

Amendments to the Drawings:

Explain all changes made to the drawings here.

Attachment: Replacement Sheet(s)
Annotated Sheet(s) Showing Changes

REMARKS/ARGUMENTS

Claims 1-28 are pending in the present application. Claims 2, 14, and 20 are canceled and claims 1, 9, 13, 18-19, 24, and 27-28 are amended. Support for the amendments to claims 1, 9, 13, 18-19, 24, and 27-28 may be located at least on page 19, lines 9-27, of the specification. The content of canceled claims 2, 14, and 20 is included in the amendments to the independent claims. Reconsideration of the claims is respectfully requested.

Amendments were made to the specification to remove the phrase "using transmission forms, such as, for example, radio frequency and light wave transmissions" on page 23, lines 10-16, as suggested by the Examiner. No new matter has been added by any of the amendments to the specification.

I. 35 U.S.C. § 101

The Office Action rejects claims 19-26 under 35 U.S.C. § 101 as being directed towards non-statutory subject matter. The Office Action states:

Claims 19-26 are not limited to tangible embodiments. In view of the applicant's disclosure, specification page 23, lines 10-16, the medium is not limited to tangible embodiments, instead being defined as including both tangible embodiments (e.g., floppy disks, RAM) and intangible embodiments (e.g., radio frequency and light wave transmissions). As such, the claim is not limited to statutory subject matter and is therefore non-statutory. Amending the specification by removing, "using transmission forms, such as, for example, radio frequency and light wave transmissions" would satisfy the requirement.

Office Action dated January 3, 2006, page 2.

In response, the specification has been amended as suggested by the Examiner. Therefore, the rejection of claims 19-26 under 35 U.S.C. § 101 has been overcome.

II. 35 U.S.C. § 103, Alleged Obviousness Based on *Arnold* and *Kolodner*

The Office Action rejects claims 1-2, 5-9, 11, 13-14, 17-20, 23-24, 26-28 under 35 U.S.C. § 103(a) as being allegedly unpatentable over *Arnold et al.* (U.S. Patent Application Publication Number 2002/0107879), hereinafter referred to as *Arnold*, in view of *Kolodner et al.* (U.S. Patent Number 6,289,360), hereinafter referred to as *Kolodner*. This rejection is respectfully traversed.

As to independent claims 1, 9, 13, 18-19, 24, and 27-28, the Office Action states:

With respect to claim 1, *Arnold* teaches of a method in a data processing system for collecting data for analyzing memory leaks, the method comprising: associating a plurality of indicators with a plurality of objects (paragraph 0048-0050),
setting an indicator for each live object in the plurality of objects to a second

state (fig. 5; paragraph 0049 and 0050; where the trace stage marks the reachable (live) objects); and

responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state (figs. 5, 6; paragraph 0050-0053; where the unreachable or unmarked object's trace record data is stored. This continues until all collectable objects trace record data is stored).

Arnold fails to explicitly teach of wherein the plurality of indicators are set to a first state. However, *Kolodner* teaches of associating a plurality of indicators with a plurality of objects, wherein the plurality of indicators are set to a first state (column 3, lines 10-25); where the colors (indicators) are associated with each object; and each object is initially unmarked or white (first state)), and

Arnold and *Kolodner* are analogous arts as they are both in the same field of endeavor, garbage collection. It would have been obvious to one of ordinary skill in the art having the teachings of *Arnold* and *Kolodner* at the time of the invention to incorporate having all objects initially being unmarked/white in marking scheme of the garbage collection taught in *Kolodner* into the garbage collection of *Arnold* as it is a common conventional method (*Kolodner* column 3, lines 8-10).

With respect to claim 9, *Arnold* teaches of a method in a data processing system for collecting data used to detect memory leaks in a Java virtual machine, the method comprising: associating an indicator with an object in a heap (fig. 4, item 46; paragraph 0036; paragraph 0048-0050);

responsive to a first request, setting indicators for all live objects in the heap from the default state to a live state (fig. 5; paragraph 0049-0050; where the trace stage marks the reachable (live) objects); and

responsive to a second request, collecting information for all objects having indicators in the default state, wherein objects having indicators in the default state are objects with memory leaks (figs. 5, 6; paragraph 0050-0053; where the unreachable object's trace record data is stored. This continues until all collectable objects trace record data is stored. As the objects that are useable are all marked, therefore the unmarked ones are not useable and thus the product of "memory leaks").

Arnold fails to explicitly teach of wherein the indicator is set to a default state. However, *Kolodner* teaches of associating an indicator with an object in a heap, wherein the indicator is set to a default state (column 3, lines 10-25; wherein the colors (indicators) are associated with each object; and each object is initially unmarked or white (default state));

With respect to claim 13, *Arnold* teaches of a data processing system in a data processing system for collecting data for analyzing memory leaks, the data processing system comprising: associating means for associating a plurality of indicators with a plurality of objects (paragraph 0048-0050; the mark-scan garbage collector marks or unmarks objects);

setting means for setting an indicator for each live object in the plurality of objects to a second state (fig 5; paragraph 0049 and 0050; where the trace stage of the garbage collector marks the reachable (live) objects); and

collecting means, responsive to a request for the data, for collecting data from all objects in the plurality of objects having indicators set to the first state (figs. 5, 6; paragraph 0050-0053; where the unreachable or unmarked object's trace record data is stored in the trace record. This continues until all collectable objects trace record data is stored).

Arnold fails to explicitly teach of wherein the plurality of indicators are set to a first state. However, *Kolodner* teaches of wherein the plurality of indicators are set to a

first state (column 3, lines 10-25; where the colors (indicators) are associated with each object; and each object is initially unmarked or white (first state)).

With respect to claim 18, *Arnold* teaches of a data processing system in a data processing system for collecting data used to detect memory leaks in a Java virtual machine, the data processing system comprising: associating means for associating an indicator with an object in a heap (fig. 4, item 46; paragraph 0036; paragraph 0048-0050; the mark-scan garbage collector marks or unmarks objects in the heap);

setting means, responsive to a first request, for setting indicators for all live objects in the heap from the default state to a live state (fig 5; paragraph 0049-0050; where the trace stage of the garbage collector marks the reachable (live) objects); and

collecting means, responsive to a second request, for collecting information for all objects having indicators in the default state, wherein objects having indicators in the default are objects with memory leaks (figs. 5, 6; paragraph 0050-0053; where the unreachable object's trace record data is stored. This continues until all collectable objects trace record data is stored in the trace record. As the objects that are useable are all marked, therefore the unmarked ones are not useable and thus the product of "memory leaks").

Arnold fails to explicitly teach of wherein the indicator is set to a default state. However, *Kolodner* teaches of wherein the indicator is set to a default state (column 3, lines 10-25; where the colors (indicators) are associated with each object; and each object is initially unmarked or white (default state));

With respect to claim 27 and 28, *Arnold* and *Kolodner* teach the limitations cited above with respect to independent claims 1 and 9. *Arnold* teaches of a bus system (fig. 4; where the processor is connected to a memory and mass storage by the busses shown in fig. 4.);

a memory connected to the bus system, wherein the memory includes a set of instructions (fig. 4; paragraph 0036-0038; where the memory is connected to the processor by the bus system. The memory contains a JVM and a program thread and collector thread (instructions)); and

a processor unit connected to the bus system, wherein the processing unit executes a set of instructions (fig. 4, paragraph 0035-0038; where the processor is connected to a memory and mass storage by the busses. The processor executes the program threads of the JVM including the collector thread which contains the garbage collector).

Office Action dated January 3, 2006, page 4-8.

As amended, claim 1, which is representative of the other rejected independent claims 13, 19, and 27 with regard to similarly recited subject matter, reads as follows:

1. A method in a data processing system for collecting data for analyzing memory leaks, the method comprising:
 - associating a plurality of indicators with a plurality of objects, wherein the plurality of indicators are set to a first state;
 - setting an indicator for each live object in the plurality of objects from the first state to a second state;
 - initiating a garbage collection process to free unused objects; and
 - responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data

associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process. (emphasis added)

As amended, claim 9, which is representative of the other rejected independent claims 18, 24, and 28 with regard to similarly recited subject matter, reads as follows:

9. A method in a data processing system for collecting data used to detect memory leaks in a Java virtual machine, the method comprising:
- associating an indicator with an object in a heap, wherein the indicator is set to a default state;
 - responsive to a first request, setting indicators for all live objects in the heap from the default state to a live state; and
 - responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks. (emphasis added)

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on the prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). For an invention to be *prima facie* obvious, the prior art must teach or suggest all claim limitations. *In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974).

Neither *Arnold* nor *Kolodner*, taken individually or in combination, teaches or suggests "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Arnold* and *Kolodner*, taken individually or in combination, do not teach or suggest "responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28.

Arnold is directed to a method, computer system and program product operable to calculate a life span of an object by determining when an object is created and becomes collectable. A garbage collection trace algorithm conducted on a method boundary reveals when the object becomes unreachable. Trace data pertaining to the collection status of the object is stored and displayed within a record file. Other data stored by the invention relates to object creation, as well as to method entry and exit. See *Arnold*, abstract. *Arnold* does not teach or suggest "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Arnold* does not teach or suggest "responsive to a second request,

collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28.

In the rejection of independent claims 1, 9, 13, 18-19, 24, and 27-28, the Office Action refers to Figure 5, Figure 6, and the following portions of *Arnold*:

[0050] When marking objects in a concurrent collection cycle, a four-color marking scheme is typically utilized. A white color indicates that an object is unmarked. A gray color indicates that an object is marked, but that its direct descendants may not yet be marked (i.e., some may be white). A black color indicates that an object is marked and all of its direct descendants are marked (either gray or black). Finally, a blue color indicates that the object is on the free list. With this scheme, gray or black objects are also referred to as "shaded" objects.

[0051] Depending upon the language being supported, additional stages may be present in a collector thread. For example, Java may require a "finalize trace" stage between the mark and sweep stages to identify and trace objects ready for finalization. A finalizer ensures that all files and resources used by object are closed prior to deallocation. The present invention can account for finalization by writing traces to a separate record, or even by using a separate deallocation algorithm to monitor the finalizer.

[0052] Once the collector locates an unreachable object at block 88, the object is identified and trace record data relating to the unreachable object is stored a block 90. For example, the invention determines that all pointers cease to reference object ins_dat at time stamp 00:02:23. The name 92 of the object is reflected in the trace record of FIG. 6, along with the collection event 94 precipitating its identification.

[0053] The JVM determines the method that was running at the time the object became collectable by associating creation and exit time stamps for respective objects and methods. The mark-scan collector continues to locate collectable objects at block 88 until all are recorded in the trace record. Then at block 96, the trace data relating to the method's exit are related to the record of FIG. 6.

Arnold, paragraphs 0050-0053.

These portions of *Arnold* disclose that a collector identifies an unreachable object and that trace record data relating to the unreachable object is stored along with other trace record data relating to object creation, method entry, and method exit, as shown in Figures 5 and 6. To the contrary, claims 1, 13, 19, and 27 recited that the data collected only contains data associated with the memory leaks. Figure 5 of *Arnold* suggests that trace record data may be stored for each instruction. Paragraph 0040 states that the invention calls for the JVM to cycle through an instruction sequence represented by Figure 5. To the contrary, the claims of the present invention recite that collecting data is responsive to a request for the data and that the data includes all objects in the plurality of objects having indicators set to the first state or default state. Additionally, the collecting steps of the independent claims occur after a garbage collection process executes to free unused objects so that the indicators only represent objects with

memory leaks. *Arnold* does not teach or suggest "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Arnold* does not teach or suggest "responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28.

Kolodner is directed to a computer-implemented method for eliminating synchronization between sweep and allocate in respect of a newly created object in a concurrent garbage collector for a heap implemented in shared memory having mark and sweep phases. In a first collection cycle, a first attribute is associated with objects believed to be reachable and a second attribute is associated with objects believed to be unreachable, whilst for each successive collection cycle, the roles of the first and second attributes are exchanged. In accordance with conventional mark-sweep garbage collectors, the attribute is a color; usually Black and White for objects which are believed to be reachable and unreachable, respectively. Exchanging the colors for each collection cycle eliminates the overhead in synchronizing the color marking of a new object depending on whether it is located in an area of the heap that has already been swept or has yet to be swept. See *Kolodner*, abstract. *Kolodner* is cited for disclosing that all objects are white at the start of the cycle since *Arnold* fails to teach or suggest that a plurality of indicators are set to a first or default state. *Kolodner* does not teach or suggest "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Kolodner* does not teach or suggest "responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28.

Arnold and *Kolodner* fail to teach or suggest "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Arnold* and *Kolodner* fail to teach or suggest "responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process

executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28. Therefore, the alleged combination of *Arnold* and *Kolodner* does not teach or suggest these features.

In view of the above, Applicants respectfully request withdrawal of the rejection of independent claims 1, 9, 13, 18-19, 24, and 27-28 under 35 U.S.C. § 103(a). Additionally, *Arnold* and *Kolodner*, taken individually or in combination, do not teach or suggest the features of dependent claims 5-8, 11, 17, 23, and 26 at least by virtue of their dependency on independent claims 1, 9, 13, 19, and 24, respectively. Claims 2, 14, and 20 are canceled. Accordingly, Applicants respectfully request withdrawal of the rejection of dependent claims 2, 5-8, 11, 14, 17, 20, 23, and 26 under 35 U.S.C. § 103(a).

Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988); *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992).

One of ordinary skill in the art would not combine *Arnold* with *Kolodner* when the references are considered as a whole. In considering the references as a whole, one of ordinary skill in the art would take into account the problems recognized and solved. The present invention recognizes the problems associated with generating data for use in detecting memory leaks particularly in a production environment. *Arnold* and *Kolodner* do not teach the problem or its source. *Arnold* is directed toward calculating a life span of an object by determining when an object is created and becomes collectable (see *Arnold*, abstract). *Kolodner* is directed toward eliminating synchronization between sweep and allocate in respect of a newly created object in a concurrent garbage collector for a heap implemented in shared memory having mark and sweep phases (see *Kolodner*, abstract). One of ordinary skill in the art would therefore not be motivated to combine or modify the references in the manner required to form the solution disclosed in the present invention.

III. 35 U.S.C. § 103, Alleged Obviousness Based on *Arnold*, *Kolodner*, and *Czajkowski*

The Office Action rejects claims 3-4, 12, 15-16, and 21-22 under 35 U.S.C. § 103(a) as being allegedly unpatentable over *Arnold* in view of *Kolodner* as applied to claims 1, 9, 13, and 19, respectively, and further in view of *Czajkowski* (U.S. Patent Number 6,594,749). This rejection is respectfully traversed.

Since claims 3-4, 12, 15-16, and 21-22 depend from independent claims 1, 9, 13, and 19, respectively, the same distinctions between *Arnold*, *Kolodner*, and the invention recited in claims 1, 9,

13, and 19 apply to dependent claims 3-4, 12, 15-16, and 21-22. In addition, *Czajkowski* does not provide for the deficiencies of *Arnold* and *Kolodner* with regard to independent claims 1, 9, 13, and 19. *Czajkowski* is directed to a system and method for memory allocation from a heap comprising memory blocks of a uniform fixed size. Each memory block has a status bit. A binary status key stores a Boolean value indicating free memory. The heap is scanned in order until a sequence of a requested quantity of free contiguous memory blocks is found or NULL is returned. Each scanned free memory block is marked un-free by assigning its status bit to the logical negative of the binary status key. If the end of the heap is reached before a sequence of sufficient quantity is found, all reachable blocks are marked as free. The binary status key is flipped such that all memory blocks which were marked free are now un-free, and vice versa. Any memory block whose corresponding structure has become unreferenced is reclaimed for future use. The scan then continues from the beginning of the heap. In another embodiment, a memory allocation for a partitioned data structure from a heap of fixed-size memory blocks may be used. The quantity of memory blocks required to store a data structure is determined. The required quantity of the memory blocks, which may be noncontiguous, is allocated from the heap. The allocated memory blocks are linked in a list such that the components of the data structure are partitioned in the proper order across the allocated quantity of memory blocks. See *Czajkowski*, abstract. *Czajkowski* is cited for allegedly disclosing that each of the plurality of indicators is a bit associated with a corresponding object in the plurality of object and that the first state is a logic zero and the second state is a logic one. *Czajkowski* does not teach or suggest "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Czajkowski* does not teach or suggest "responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28. Thus, any alleged combination of *Czajkowski* with *Arnold* and *Kolodner* still would not result in the invention recited in claims 1, 9, 13, and 19 from which claims 3-4, 12, 15-16, and 21-22 depend. Accordingly, Applicant respectfully requests withdrawal of the rejection of claims 3-4, 12, 15-16, and 21-22 under 35 U.S.C. § 103(a).

IV. 35 U.S.C. § 103, Alleged Obviousness Based on *Arnold*, *Kolodner*, and *Adl-Tabatabai*

The Office Action rejects claims 10 and 25 under 35 U.S.C. § 103(a) as being allegedly unpatentable over *Arnold* in view of *Kolodner* as applied to claims 9 and 24, respectively, and further in

view of *Adl-Tabatabai et al.* (U.S. Patent Number 6,317,869), hereinafter referred to as *Adl-Tabatabai*. This rejection is respectfully traversed.

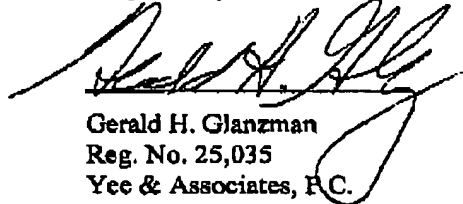
Since claims 10 and 25 depend from independent claims 9 and 24, respectively, the same distinctions between *Arnold*, *Kolodner*, and the invention recited in claims 9 and 24 apply to dependent claims 10 and 25. In addition, *Adl-Tabatabai* does not provide for the deficiencies of *Arnold* and *Kolodner* with regard to independent claims 9 and 24. *Adl-Tabatabai* is directed to a method for run-time tracking of object references in computer code and determining which variables contain references to objects at garbage collection sites. The method of the present invention first creates a bit vector in memory. The bit vector is then initialized. Second, each variable declared in the computer program that may be used to store a reference value is assigned a unique bit within this bit vector. Each bit is maintained to indicate whether the variable it is assigned to is currently storing a reference value. Specifically, when a variable is assigned a reference value, the corresponding bit in the bit vector is set. When a variable is assigned a non-reference value, the corresponding bit in the bit vector is cleared. See *Adl-Tabatabai*, abstract. *Adl-Tabatabai* is cited for allegedly disclosing that an indicator is associated with the object when the object is created. *Adl-Tabatabai* does not teach or suggest "responsive to a request for the data, collecting data from all objects in the plurality of objects having indicators set to the first state, wherein the data collected only contains data associated with the memory leaks and wherein the memory leaks occur when unused objects are not freed during the garbage collection process," as recited in claims 1, 13, 19, and 27. Additionally, *Adl-Tabatabai* does not teach or suggest "responsive to a second request, collecting information for all objects having indicators in the default state after a garbage collection process executes to free unused objects, wherein objects having indicators in the default state are objects with memory leaks," as recited in claims 9, 18, 24, and 28. Thus, any alleged combination of *Adl-Tabatabai* with *Arnold* and *Kolodner* still would not result in the invention recited in claims 9 and 24 from which claims 10 and 25 depend. Accordingly, Applicant respectfully requests withdrawal of the rejection of claims 10 and 25 under 35 U.S.C. § 103(a).

V. Conclusion

It is respectfully urged that the subject application is patentable over the cited references and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: April 3, 2006

Respectfully submitted,



Gerald H. Glanzman
Reg. No. 25,035
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants

GHG/vja